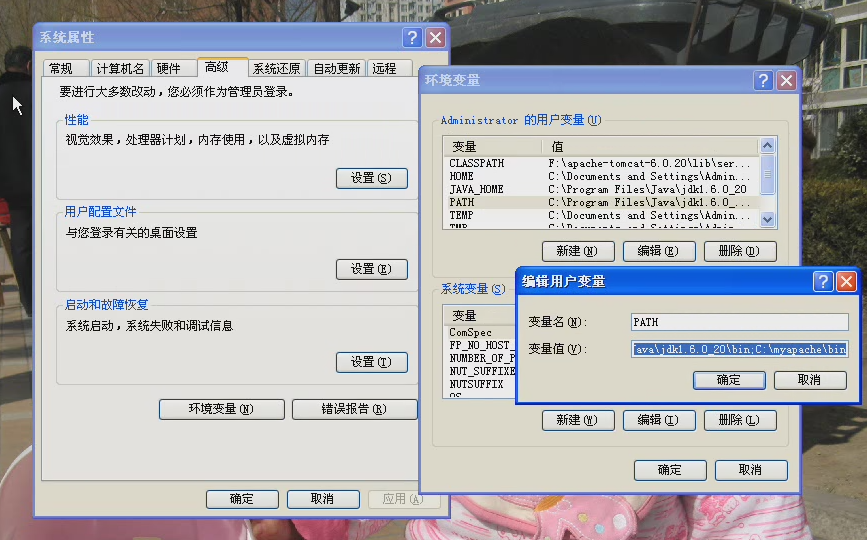
**PHP**

PHP= PHP: Hypertext Preprocessor

**命令行管理apache：**

1. 运行 CMD
2. 输入 cd D:\wamp\apache\bin （注：cd的意思是切换到后面的路径目录）
3. 输入 httpd –k start(or restart or shutdown)

如果希望任何目录都能运行httpd，则需设置PATH环境变量（28讲20分钟）：



将bin路径粘贴到变量值的最后面

**端口（port）：**

1. 一台机器可以有1-65535号端口
2. 在实际开发中，我们经常使用CMD内netstat –an 来查看机器有哪些端口在监听

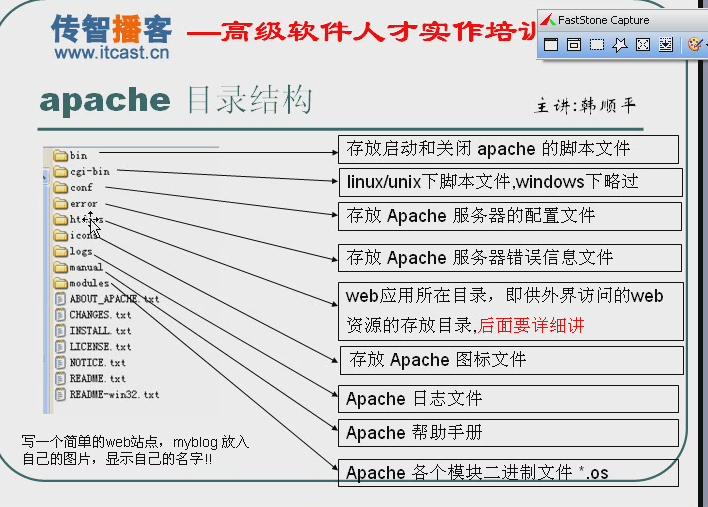
端口越少越安全

1. 如果发现有异常端口监听，则可以关闭该端口，在任务管理器的进程里关闭
2. 一台机器的一个端口被一个程序监听了，则该端口不能再被别的应用程序监听
3. 端口分为有名端口 1-1024 号（这些端口别随意使用），其它端口可自行分配

**apache的配置在httpd.conf 文件中，路径是：D:\wamp\apache\conf 需重启生效**

修改端口：Listen 80 可添加多个端口Listen 81 Listen 82

**apache目录解释：**



比较重要的文件夹：bin conf htdocs modules

**配置虚拟目录：**

1. 添加虚拟目录节点：在以下位置添加



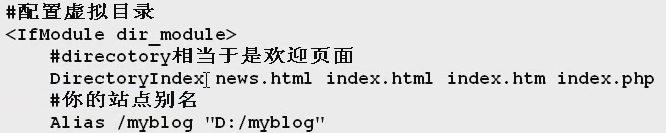
1. 注销documentroot路径：在前面加个井号即可（可不注销）



1. 测试

在浏览器输入新路径地址：<http://localhost/myblog/news.html>

1. 配置欢迎页面：以下位置添加news.html



在浏览器输入新路径地址：<http://localhost/myblog/> 测试

**配置虚拟机：**

1、在httpd.conf 中找到 Virtual hosts段落，将下面一行的注释符号去掉以启用



2、在extra文件夹里的httpd-vhosts.conf 文件里配置

在文件末尾加上

这里配置自己虚拟主机的IP



可在这里添加首页面，比如自己创建在myblog里的页面example.html，那么在地址栏中不用输入example.html也会访问到这个页面

这里的路径需要自己设置，即创建的站点的路径

**\*29讲 20分钟有关于网络访问流程的详细解释**

**\*以上的路径在修改的时候如果直接粘贴复制路径，需要将\改为/**

3、在hosts中添加我们的IP和主机映射关系，如：



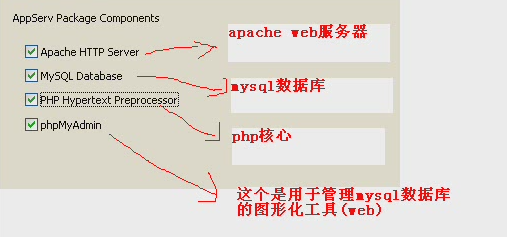
前面的127.0.0.1就是上面配置虚拟主机里自己设置的IP

那么在浏览器中输入[www.sohu.com](http://www.sohu.com)就能浏览自己创建的站点

**\*apache在默认情况下不会处理PHP页面，也就是输入PHP页面的路径也无法打开，而是提示下载，需要相关配置后才能处理**

**PHP开发环境的搭建**

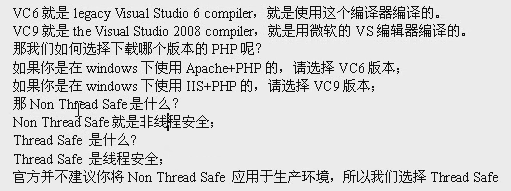
1、安装WAMP套件，套件里的包含apache mysql php phpmyadmin，作用是：



2、自定义安装各个部件

①下载apache 完成后输入<http://localhost> 测试

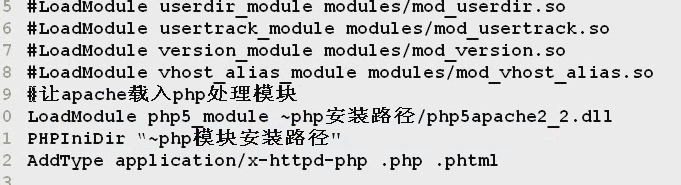
②下载php核心包，分为VC6和VC9两个版本

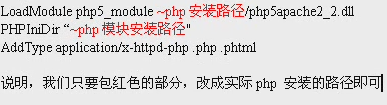


PHP不用安装，解压到安装目录即可

③如何将apache和php整合

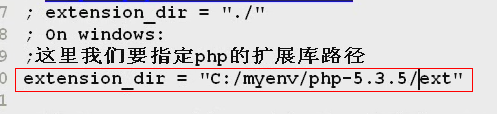
在apache的conf目录下的httpd.conf的LoadModule下加入如下代码





把 php.ini-development 文件改成php.ini，因为PHP的设置需要在此文件修改

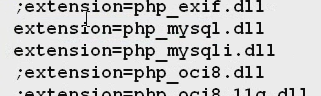
在php.ini中可以指定对应的功能模块，在里面指定PHP的扩展库路径



注意在复制路径的时候把\改为/

④安装mysql

安装完后在php.ini中启用mysql



去掉前面的注释符号；

⑤安装phpmyadmin

下载后解压到apache的htdocs目录下，然后直接访问<http://localhost/phpmyadmin/index.php>即可

**在apache里创建多个站点：**



只需要多添加一个虚拟主机，然后添加ServerName，即访问域名，并在host里再添加一个地址即可



**PHP基本语法**

1. PHP可以和HTML混编，基本格式<html><?php …… ?></html>
2. 必须以”;“结束
3. PHP变量区分大小写，值不区分

**\*编程就是创造一个世界，数据类型好比“人、山、草、木” 变量好比“张三、旺财、某张桌子” 数据类型是一个大类，变量是类型中的一个具体实物，好比变量“张三”属于类型“人”**

<?php

$a=100

$a=50

?>

$是定义变量的符号，后面的定义会覆盖前面的，所以结果是a=50

**整型细节讨论：**

$a的整型赋值可以是十进制，八进制，十六进制

$a=0123 //八进制（通常以0开头）

换算方法：0123=3\*8的0次方+2\*8的1次方+1\*8的2次方=3\*1+2\*8+1\*64=83

&a=0x1A //十六进制

换算方法：0x1A=A\*16的0次方+1\*16的1次方=10\*1+16=26

**布尔值：**

除了false外，以下值都会被当作false对待：

0、0.0、空字符串、字符串‘0’，不包括任何元素的数组，不包括任何成员变量的对象（仅PHP4.0）、特殊类型null（包括尚未设定的变量）

**浮点型：**

浮点数的精度是14位，精度是指从左边开始，第一个非0数开始计算

**字符串：**

一个字符串占用一个字节，字符串大小的最大值就是计算机的内存大小

“”和‘’的区别：

双引号内的字符串，如&a，如果前面把&a定义为了一个变量，则字符串内的&a会替换为变量值，而单引号则会正常打印出&a

**\*要看变量的类型使用 var\_dump(变量); 来查看**

**\*表达式：所谓表达式，就是任何有值的东西**

**算数运算符：**

+、-、\*、/、%（取模，表示求余数）

1、++和--，++分为前++和后++，--也一样

前++和后++的区别：

$a=56;

$b=++$a; //等价于：$a=$a+1; $b=$a;

$c=$a++; //等价于：$c=$a; $a=$a+1;

acho $b; //这个的值是：57

acho $c; //这个的值是：56

写在前面就是先++再判断，写在后面就是先判断再++

2、+=，-=，\*=，/=，%=

$a=90;

$a+=90; //等于$a=$a+90

acho $a; //值是180 别的符号同理

**逻辑运算符：**



and和or的优先级很低，比=还低

**三元运算符：**

**基本语法：**

表达式1？ 表达式2：表达式3

规则：如果表达式1的运算结果是true，则取表达式2的值，否则取3的值

$a=90;

$b=80;

$c=$a>$b? (12-5):“你好”

echo “c=”.$c //则打印出：c=7

**类型运算符：**

基本语法是：instanceof 用于确定一个PHP变量是否属于某一类class的实例

class dog{}

class cat{}

//创建一个对象

$cat1=new cat;

var\_dump($cat1 instanceof dog); //则会输出bool(false),因为cat1应该属于cat类

**PHP三大流程控制**

**1、顺序控制**

这个就是基本的从上到下的流程

**2、分支控制**

有选择的来执行我们的代码

分为单分支，双分支，多分支

**单分支语法：**比如if语句

$age=25;

if($age>18){

echo’你的年龄大于18‘

}

**双分支语法：**

$age=25;

if($age>18){

echo’你的年龄大于18‘

}else｛

echo’你的年龄小于18‘

｝

**多分支语法：**

if（条件表达式）｛

//n多语句

｝else if（条件表达式）｛

//n多语句

｝//这里可能还有更多else if

else｛

｝

注：else if可以有一个，也可以有多个；末尾的else可以没有

案例：

$age=2;

if($age>18){

echo’你要被送入监狱’;

}else if($age>10 && $age<=18){

echo’送入少管所’;

}else{

echo’下次注意’;

}

**switch分支语句：**

基本语法结构：

switch（表达式）｛

case 常量1：

//n多语句；

break;

case 常量2：

//n多语句；

break；

default:

//n多语句

break；

｝

注意：1、case语句有一到多个 2、default语句可以没有（根据自己代码的业务逻辑）

4、通常，在case语句后要带break，表示退出switch语句4、常量的类型（int/float/字符串/布尔型）

//当这里的值是a的时候，会转换为下面a的语句跳过后面的，是b的时候转换为b，跳过后面的，abc都不是则转换为最后一句

$myChar=”a”;

switch{myChar}{

case”a”:

echo ‘今天星期一，猴子穿新衣’;

//break的作用是终止运算，如果没有break，即使上面是a也会输出abc，如果上面是b，会输出bc

break；

case“b”：

echo‘今天星期二，猴子当小二’；

break；

case“c”：

//default的位置可以在开头或中间，执行的顺序是：首先按照case顺序进行匹配，如果一个都没匹配到，则匹配default的内容，直到遇到break退出，否则依然会输出default下面的内容；如果匹配到了，即使default在前面，也只输出匹配到的那个内容

echo‘今天星期三，猴子爬雪山’；

break；

default：

echo‘没有匹配的日期’；

｝

**\*if和switch的区别：if通常是用于范围，switch用于一个点**

**循环控制**

**for循环控制**

基本语法

for（循环初值；循环的条件；步长）｛

//n多语句；

｝

例：for($i=0;$i<10;$i++){

echo ‘hello，world‘；

｝ //输出10句hello,world

**while循环**

while（循环条件）｛

//循环体，语句

｝

例：$i=0; //$i是循环控制变量

while（$i<10）{

echo ‘hello,world’;

$i++;

}

**do…while循环**

do{

//循环体

}while(循环条件);

例：$i=0;

do{

echo ‘hello,world’;

$i++; //这里由程序员自己控制步长，比如可以改为$i=$i+3

}while($i<10);

**循环相关语句-break**

表示结束当前的for/while/do..while/switch流程；可以在后面加上一个数字，表示退出几层，如：break 2，默认是1

$i=0;

while(++$i){

switch($i){

case 5;

echo:’quit at 5<br/>;

break;

case10;

echo:’quit at 10<br/>’;

break 2;

default;

break;

}

}

echo ‘$i=’.$i;

将输出：

quit at 5

quit at 10

$i=10

**循环相关语句-continue**

用于结束本次循环剩余代码，重新开始新的一次循环，如果条件为真就继续执行；continue后面也可以带数字，表示从第几层循环重新开始

for($i=0;$i<3;$i++){

for($j=1;$j<5;$j++){

if($j==3){

continue 2;

}

echo ‘$i=’.$i.’$j’=$j.’<br/>’;

}

输出：

$i=0$j=1

$i=0$j=2

$i=1$j=1

$i=1$j=2

$i=2$j=1

$i=2$j=2

\*注：多层循环语句，是要先运行内层的循环，直到break或continue后才会终止跳到上一层循环。

**goto语句**

通过goto语句我们可以将程序跳转到指定的地方去执行

语法：

goto 标签；

标签：

语句；

例：goto a;

echo “abc”;

a:

echo “123”;

将输出：123

goto语句某些时候可用于while或switch中，代替break跳出

**常量**

常量可以理解成就是特殊的变量；体现在：

1. 定义常量前面不需要$
2. 常量一旦定义，则不能修改其值
3. 常量在定义的时候就需要给它赋初值
4. 常量可以通过define()或者const定义
5. 常量的名称一般是全部大写，用下划线间隔
6. 什么时候需要使用常量：在程序中我们不希望某个值变化，则考虑使用常量

例：

1. define(“FIRST\_NAME”,”He”);

echo FIRST\_NAME;

1. const AGE=25;

echo AGE;

**函数**

**为了完成某一功能的程序指令的集合，称为函数；就是一个功能块；函数包括自定义函数和系统函数**

引入函数的方法：在需要引入的PHP文件中使用require引入含有函数的PHP文件

<?php

require ‘function.php’;

?>

例：我们制定一个计算+-\*/的代码集合

1. 函数文件function.php

<?php

//1.function是一个关键字

//2.jiSuan是函数名（由程序员自己取）

//3.$num1,$num2,$oper是函数的参数列表（形参）

function jiSuan($num1,$num2,$oper){

//$res表示计算的结果

$res=0;

switch($oper){

case’+’

$res=$num1+$num2;

break;

case’-’

$res=$num1-$num2;

break;

case’\*’

$res=$num1\*$num2;

break;

case’/’

$res=$num1/$num2;

break;

default;

echo’运算符号有误’;

}

return $res;

}

?>

1. 执行文件run.php，这个文件需要引入function.php

<?php

require’function.php’;

$num1=10;

$num2=25;

$oper=”+”;

$res=jiSuan($num1,$num2,$oper);

echo’计算结果是=’.$res;

?>

**自定义函数语法结构：**

function 函数名（参数列表）｛ //参数列表用于接收数据

//函数体；（完成某一功能的指令集合，必须有）

//return语句；（主要用途是返回一个结果，不是必须）

｝

**PHP的引入：**

可用require和require\_once引入

require可以多次引入同一个php文件；require\_once会判断，如果前面有引入相同的php文件，则不再引入了

**在一个PHP中，不能有两个函数名相同的函数**

include和include\_once的使用方法和require一样，不同之处是：require遇到错误会终止执行，而include遇到错误会继续往下执行

**在做项目的时候基本上都是使用require\_once，而且应当放在PHP最前面，先引入**

**函数的运行状况**

**按照函数执行的规则，只要一看到函数，PHP就会创建一个新栈（每个函数都是一个独立的空间）；各个栈里的变量是相互独立的，不同栈里的变量可以是相同的变量名**

<？php

function abc($n){

if($n>2){

abc(--$n);

}

echo’$n=’.$n.”<br/>”;

}

abc(4);

?>

将输出：

$n=2

$n=2

$n=3

**函数的基本结构**

1. 参数列表可以是多个
2. 数据类型可以是任意类型（array,integer,float,Boolean,string,object,null,资源类型）
3. 函数的命名规则和自定义变量基本一样，但函数名不区分大小写
4. 使用global全局变量的时候，可以使用在函数外的变量（global会把函数外的变量引入进函数）

<?php

$a=12

function abc(){

global $a;

$a+=45

}

abc();

echo $a;

?}

这样将输出57，如果没有global将输入12

1. 如果在函数中，我们不希望使用某个变量，或者是希望彻底的不再使用某个变量，则可以使用unset(变量名)；将该变量彻底删除
2. 在PHP函数中，我们可以给某些参数赋一个默认值

function abc($b,$a=2){ //这里给$a赋予了一个默认值

$res=$a+$b;

return $res;

}

$e=70;

echo abc($e); //由于已经给$a赋了默认值，所以这里没第二个值也不会报错

echo abc($e,90); //这里将会用90替代$a的默认值2

1. PHP传递变量的时候，默认是值传递，如果需要引用（地址）传递，可以使用&变量名

$a=213;

function abc(&$b){ //这里将替换$a

$b=333;

}

abc($a);

echo $a;

将输出：333

**位运算**

**二进制**

所谓二进制就是逢二进一。为什么要用二进制：因为在二进制中只有0和1两个数，非常简单方便，易于用电子方式实现。

**原码、反码、补码**

原码：用二进制来表示一个数，这个码就是原码

如：00000000 00000000 00000000 01101010

1. 二进制的最高位是符号位：0表示正数，1表示负数
2. 正数的原码、反码、补码都一样
3. 负数的反码=它的原码符号位不变，其它位数取反：0→1，1→0

如：-1的原码：10000000 00000000 00000000 00000001

-1的反码：11111111 11111111 11111111 11111110

④ 负数的补码=它的反码+1

如：-1的原码：10000000 00000000 00000000 00000001

-1的补码：11111111 11111111 11111111 11111111

⑤ 0的补码、反码都是0

⑥ PHP没有无符号的数

⑦ 在计算机运算的时候，都是以补码的方式来运算的

**基本概念**

PHP中有4个位运算，分别是：按位与&、按位或|、按位异或^、按位取反~

运算规则是：

按位与&：两位全为1，结果为1

按位或|：两位有一个为1，结果为1

按位异或^：两位一个为0，一个为1，结果为1

按位取反~：0→1，1→0

例：

1、~2=？

步骤：①求出2的补码：00000000 000000000 00000000 00000010

②取反：11111111 11111111 11111111 11111101（此结果并非最终结果，因为计算机是以补码方式结算的，所以我们还需要求出原码）

③补码→反码：11111111 11111111 11111111 11111100

④反码→原码：10000000 00000000 00000000 00000011

⑤最终结果：-3

2、2&3=？

步骤：①求出2和3的补码

2的补码：00000000 000000000 00000000 00000010

3的补码：00000000 000000000 00000000 00000011

②求按位与：2&3：00000000 00000000 00000000 00000010

③求原码：由于是2&3是正数，所以原码就是上面的结果

④最终结果：2

**位移运算**

>>算术右移；<<算术左移

运算规则：

算术右移>>：低位溢出，符号位不变，并用符号位补溢出的高位

算术左移<<：符号位不变，低位补0

例：

1、$a=1>>2

1的补码：00000000 00000000 00000000 00000001

1>>2：00000000 00000000 00000000 00000000

如何运算的：1的补码往右移动2位：0？？00000 00000000 00000000 00000000

再用符号位补上溢出的高位：00000000 00000000 00000000 00000000

2、$a=-1<<2

-1的原码：10000000 00000000 00000000 00000001

-1的反码：11111111 11111111 11111111 11111110

-1的补码：11111111 11111111 11111111 11111111

1<<2：11111111 11111111 11111111 11111100

补码→反码：11111111 11111111 11111111 11111011

反码→原码：10000000 00000000 00000000 00000100

**数组**

可以存放多个数据的数据类型，我们称为数组

在PHP数组中，元素存放的值可以是任意数据类型

**创建数组**

1、在PHP中，数组就是关键字和值的集合，我们可以这样创建

$array[0]=123

$array[1]=”soul”

$array[2]=25.7

2、$数组名=array(值……);

$array=array(10,”默殇”,true);

遍历数组：

for($i=0;$i<count($array);$i++){ //count()是系统该函数，作用是统计数组个数

echo $array[$i].“<br/>”;

}

3、自己指定下标创建方式

$arr[‘name’]=’默殇’;

$arr[‘age’]=25;

……

或者：

$arr=array(‘name’=>’默殇’,’age’=>25,……);

这样指定后需要使用foreach函数来遍历数组：

foreach($arr as $key=>$val){ //$key和$val的名称可以自己命名

echo $key.”=”.$val.”<br/>”;

} //当然，这也可以用于普通数组

我们可以使用print\_r或var\_dump来显示数组的具体情况：

$array=array(123,’soul’,25);

print\_r($array);

var\_dump($array);

php的数组可以动态增长：

$a=array(23,”hello”);

$a[3]=10;

echo $a[3];

**数组的几个重要函数**

1. count函数

可以统计该数组共有多少元素

基本用法是：count($数组名);

1. print\_r和var\_dump函数

可以显示数组的具体情况

基本用法是：print\_r($数组名); var\_dump($数组名);

1. is\_array函数

可以判断是否是数组

基本用法是：is\_array($名称);

如：echo is\_array($arr);显示1则表示是数组，如果是空则表示不是数组

1. explode函数

主要用于拆分字符串

基本用法是：$数组名=explode(“用于拆分的关键字”,拆分的变量);

如：$str=”利物浦&曼联&切尔西”;

$arr=explode(“&”,$str);

print\_r($arr);

1. unset函数

可以销毁某个元素，也可以销毁某个变量

基本用法：unset($arr[2]);或unset($a);这样就删除了此元素或变量

**数组运算符**



$a=array(“a”=>”apple”,”b”=>”banana”);

$b=array(“a”=>”pen”,”b”=>”eraser”,”c”=>”book”);

$c=$a+$b;

var\_dump($c); //结果是：”a”=>”apple”,”b”=>”banana”,”c”=>”book”

**数组的排序**

排序是将一组数据，按照某个顺序进行排列

排序是一个程序员的基本功，一些常用的排序方法

排序分两大类：内部排序法（在内存中排序）和外部排序法（内存不足，在硬盘中排序）

以下都是内部排序法

1. 冒泡
2. 选择排序法
3. 插入排序法
4. 快速排序法

**查找**

1. 顺序查找  
   对某个数组，按照顺序，一个一个比较，然后找到你要的数据  
   例：要求从一个数组：$arr=array(46,90,900,0,-1)中查找一个数34，如果找到则输出该数的下标；如果找不到，则输出查无此数  
   $arr=array(46,90,900,0,-1,5,90,8)  
   function search(&$arr,$findVal){

$flag=false;  
 for($i=0;$i<count($arr);$i++{

if($findVal==$arr[$i]){

echo”找到了，下标为=$i”;

$flag=true;

//break; （有break的时候只能找到一个，没有可以找到多个）

}

}

if(!$flag){

echo’查询不到’;

}

}

search($arr,34);

1. 二分查找

所谓二分查找，它有一个重要前提，该数组本身已经是一个有序数组，如果该数组不是有序的，则必须先排序再查找。  
首先找到数组中间这个数，然后与要查找的数比较，如果要查的数大于中间这个数，则说明应该向后找，否则向前找；如果相等，则说明找到了。

arr=array(-40,-10,-8,2,10,18,60,74);  
function binarySearch(&$arr,$findVal,$leftIndex,$rightIndex){

//当$leftIndex>$rightIndex说明没有该数

if($leftIndex>$rightIndex){

echo”找不到该数”;

return;

}

//找到中间这个数

$middleIndex=round(($rightIndex+$leftIndex)/2); //round是四舍五入

//如果大于，则向后找

if($findVal>$arr[$middleIndex]){

binarySearch($arr,$findVal,$middleIndex+1,$rightIndex);

}

//如果是小于中间这个数，则向前找

else if($findVal<$arr[$middleIndex]){

binarySearch($arr,$findVal,$leftIndex,$middleIndex-1);

}

//找到了

else{

echo”找到这个数，下标是$middleIndex”;

}

binarySearch($arr,10,0,count($arr)-1);

//10是查找的数，0和count($arr)-1表示最小下标和最大下标

**多维数组·二维数组**

基本语法

$arr=array(array(1,2,3),array(“hello”,”world”),array(5,6,7));

要访问“hello”，代码为$arr[1][0]

例：用二维数组输出如下数据：

0 0 0 0 0 0

0 0 1 3 4 8 2

3 7 8 3 9

2 9 0 7 3 7

//先定义一个数组

$arr=array(

array(0, 0, 0, 0, 0, 0),

array(0, 0, 1, 3 ,4, 8, 2),

array(3, 7 ,8, 3, 9),

array(2, 9, 0, 7, 3, 7));

//遍历二维数组（行）

for($i=0;$i<count($arr);$i++){

//遍历每行的数据

for($j=0;$j<count($arr[$i]);$j++){

echo $arr[$i][$j].”&nbsp;”;

}

echo”<br/>”;

}

**面向对象编程**

**包括类和对象，类是抽象的，比如：人类；对象是具体的，比如：身高，体重，年龄。**

**创建类：**

class Person{ //类名要求首字母大写

public $height;

public $weight;

public $age;

}

**创建对象：**

$p1=new Person();

$p1->height=180;

$p1->weight=160;

$p1->age=30;

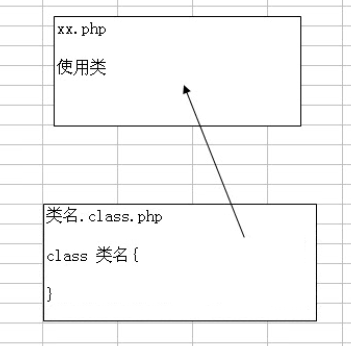
成员属性的说明

1. 成员属性是从某个事物提取出来的，它可以是基本数据类型（整数、小数、字符、布尔），也可以是复合数据类型（数组、对象）

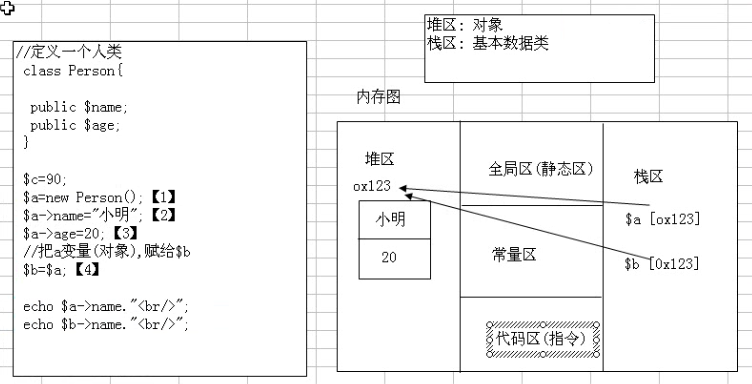
如何创建对象：

1. 怎样访问对象属性：$对象名->属性名;

\*通常在定义类的时候是单独创建一个php文件，文件的命名方式一般是：类名.class.php

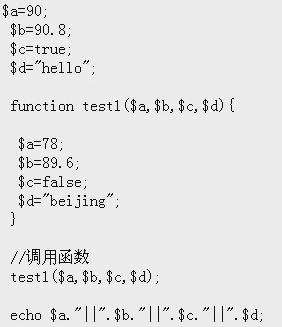


**对象的内存图**



**注：**1、给函数传递的是对象，则传入的是地址而不仅仅是变量的值

2、给函数传递的是基本数据类型，则传入的只是值而不是地址，所以在返回的时候并不会修改变量的值



输出的依然是：90、90.8、1、hello

如果$a…改为&$a，则传入的是地址，输入结果便会改变为函数内的值。

3、给函数传递的是数组，则默认情况下传入的是值，如果希望传入地址，则应当加上地址符：&$数组名

**成员方法（函数）**

函数和成员方法的关系：当我们把一个函数写到某个类中，则该函数可以成为成员方法

方法的作用是赋予类的作用和行为，比如人类可以说话、跑步、做题等

<?php

class Person{

public $name;

public $age;

public function speak(){ //其实就是把函数的位置放到了类中

echo’我是一个函数’

public function count1(){

$res=0;

for($i=1;$i<=1000;$i++){

$res+=$i;

}

//return究竟到什么地方去？谁调用，就返回给谁

return $res;

}

//修改成员方法，该方法可以接收一个数n，计算1+……+n的结果

public function count2($n){

$res=0;

for($i=0;$i<=$n;$i++){

$res+=$i;

}

return $res;

}

public function add($num1,$num2){

return $num1+$num2;

}

}

//如何使用函数

//1、创建一个人

$p1=new Person();

//2、通过对象去访问（调用）成员方法

$p1->speak();

//3、让这个人做算术题

$res=$p1->count1();

echo’计算结果是=’.$res;

//4、让这个人可以接收一个数，然后计算

echo ‘计算结果是=’.$p1->count2(50);

//5、让这个人计算两个数的和

echo’50+20=’.$p1->add(50,20);

?>

**构造方法（函数）**

需求：在创建一个人对象的时候，就初始化name和age

class Person{

public $name;

public $age;

//构造方法

public function\_\_construct($iname,$iage){ //PHP5才有，并优先选择

$this->name=$iname; //系统会被每个对象分配this，代表当前对象

$this->age=$iage; //$this只能在类里面使用

echo”我是构造方法”;

}

//PHP4中的构造方法，名字和类名一致

public function Person(){ //这个不建议使用

echo”OK”

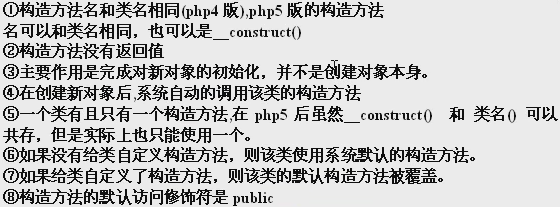
}

}

$p1=new Person(“宋江”,35);

echo $p1->name;

构造方法的特点：



**析构方法（函数）**

主要作用：释放资源（比如释放数据库的链接，图片资源，销毁某个对线等等）这样就能回收内存、CPU等

class Person{

public $name;

public $age;

//构造方法

public function \_\_construct($name,$age){

$this->name=$name;

$this->age=$age;

//这里会打开一个链接库的资源

}

//写一个析构方法 \_\_是两个下划线

function \_\_destruct(){

echo $this->name.“销毁资源 关闭数据库<br/>”;

}

}

$p1=new Person(“贾宝玉”,16);

$p2=new Person(“林黛玉”,14);

运行代码，我们得出：

1、析构函数会自动调用

2、析构方法主要用于销毁资源

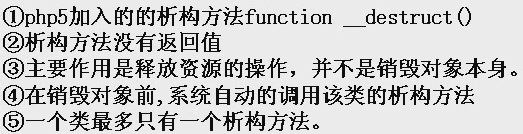
3、析构函数调用的顺序是，先创建的对象后被销毁（栈先入先出）

4、析构函数什么时候被调用？

①当程序（进程结束）退出时

②当一个对象成为垃圾对象的时候，该对象的析构函数也会被调用  
③所谓垃圾对象是指没有任何变量在引用它  
④当一个对象成为垃圾对象，析构函数就会立即调用

析构函数小结：



**静态变量（类变量）和静态方法（类方法）**

所有对象都共享的一个变量叫静态变量。

例：有一群小孩在玩堆雪人，不时有新的小孩加入，请问如何知道现在共有多少人在玩？

1. **使用全局变量**  
   什么是全局变量？在程序中，任何地方都可以使用的变量称为全局变量  
   <?p  
    //定义一个全局变量  
    global $a;  
    //给全局变量$a赋值  
    $a=9;  
    function test1(){  
    global $a;  
    $a=90;  
    //调用了函数  
    test1();  
    echo $a;  
   ?>

例子解：

<?php

//定义 玩游戏的全局变量

global $global\_nums;

//赋值

$global\_nums=0;

class Child{

public $name;

function \_\_construct($name){

$this->name=$name;

}

public functione join\_game(){

//声明一下使用全局变量

global $global\_nums;

$global\_nums+=1;

echo $this->name.”加入堆雪人游戏”;

}

}

//创建3个小孩

$child1=new child(“李逵”);

$child1->join\_game();

$child2=new child(“张飞”);

$child2->join\_game();

$child3=new child(“唐僧”);

$child3->join\_game();

//看看有多少人玩游戏

echo “<br/>有”.$global\_nums;

?>

1. **使用静态变量**

如何定义静态变量：

访问修饰符 static 变量名;

如何访问静态变量：

在类外部：类名::$静态变量名

在类内部：self::$静态变量名 或者 类名::$静态变量名  
例子解：

<?php

class Child{

public $name;

//定义并初始化一个静态变量 $nums

public static $nums=0;

function \_\_construct($name){

$this->name=$name;

}

public functione join\_game(){

//使用静态变量

self::$nums+=1;

echo $this->name.”加入堆雪人游戏”;

}

}

//创建3个小孩

$child1=new child(“李逵”);

$child1->join\_game();

$child2=new child(“张飞”);

$child2->join\_game();

$child3=new child(“唐僧”);

$child3->join\_game();

//看看有多少人玩游戏

echo “<br/>有”.Child::$nums;

?>

1. **静态方法（函数）**

需求：当我们操作静态变量的时候，我们可以考虑使用静态方法

\*静态方法不能操作非静态变量

例：统计所有学生交的学费

<?php

class Student{

//创建静态变量

public static $fei;

//创建非静态变量

public $name;

//构造函数

function \_\_construct($name){

$this->name=$name;

}

//入学

public static function enterSchool($ifei){

self::$fei+=$ifei

}

//获取总学费

public static function getFei(){

return self::$fei;

}

}

//创建学生

$stu1=new Student(“小倩”);

//1.通过类名来调用静态方法

//Student::enterSchool(340);

//2.通过对象名->类方法名

$stu1->enterSchool(340);

$stu2=new Student(“小采”);

$stu2->enterSchool(300);

echo”总学费是=”.Student::getFei();

?>

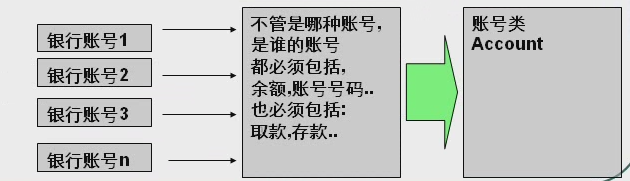
静态方法的特点：

1. 静态方法去操作静态变量
2. 静态方法不能操作非静态变量（注：普通的成员方法既可以操作非静态变量，也可以操作静态变量）

**PHP面向对象编程的三大特征**

三大特征：封装 继承 多态

**抽象：**我们在前面去定义一个类的时候，实际上就是把一类事物的共有的属性和行为提取出来，形成一个物理的模型（模版）。这种研究问题的方法称为抽象。研究问题的方法称为抽象。



**封装：**就是把抽象出的数据和对数据的操作封装在一起，数据被保护在内部，程序的其它部分只有通过被授权的操作（成员方法），才能对数据进行操作。如对电视机的操作就是典型的封装。

PHP提供了三种访问控制修饰符：

public 表示全局 类内部、外部和子类都可以访问 //默认值

protected 表示受保护的 只有本类或子类可以访问

private 表示私有的 只有本类内部可以访问

\*这三个控制修饰符号可以对属性和方法修饰

1.如果一个方法没有访问修饰符，则默认是public

2.属性必须指定访问修饰符